

Faculty of Information Technology

Computer Systems Engineering Department

COMPUTER DESIGN LAB #ENCS411

EXP #8

Programmable Interval Timer

MDA-8086 Kit – PPI Application

**Name : Mohammad Modallal**

**ID : 1120174**

**Section# : 3**

**Dr.** [**Khader Mohammad**](https://ritaj.birzeit.edu/student/view_office_hr?employee_id=5442)

**Date:27/4/2015**

1. Contents

[1 Abstract 3](#_Toc387987979)

[2 Introduction to PIT 4](#_Toc387987976)

[2.1 Defnition of PIT 4](#_Toc387987981)

[2.2 Features 4](#_Toc387987982)

[2.3 Examples of usage 4](#_Toc387987983)

[2.4 8253/4 Functional Description 5](#_Toc387987984)

[2.5 Programming the 8254 6](#_Toc387987984)

[3 Procedure 7](#_Toc387987989)

[3.1 Configuring PIT on PC 7](#_Toc387987990)

[3.2 assembly program 8](#_Toc387987991)

[3.3 Configuring PIT on MDA-8086 Kit 9](#_Toc387987991)

[3.4 increase the delay by a factor of 5 times 11](#_Toc387987991)

[4 Conclusion 14](#_Toc387987992)

[5 References 15](#_Toc387987992)

**1.Abstract**

The aim of this experiment is to understanding , configuring and testing the Programmable Interval Timer (PIT) devices[ 8253/4] on the MDA\_8086 kit and Personal Computers.

**2.Introduction**

**2.1 Definition**

The [Intel](http://en.wikipedia.org/wiki/Intel) 8253 /4 are [Programmable Interval Timers](http://en.wikipedia.org/wiki/Programmable_Interval_Timer) (PITs), which perform timing and counting functions. They were primarily designed for the [Intel 8080](http://en.wikipedia.org/wiki/Intel_8080)/[8085](http://en.wikipedia.org/wiki/Intel_8085)-processors, but later used in x86-systems. They (or an equivalent circuit embedded in a larger chip) are found in all [IBM PC compatibles](http://en.wikipedia.org/wiki/IBM_PC_compatible)[[1]](http://en.wikipedia.org/wiki/Diode#cite_note-5).

**2.2 Features**

The programmable interval timer consists of three independent 16-bit programmable counters (timers). Each counter is capable of counting in binary or BCD[[2]](http://en.wikipedia.org/wiki/Diode#cite_note-5).

**2.3** **Examples of usage**[[2]](http://en.wikipedia.org/wiki/Diode#cite_note-5)

The programmable interval timer is useful wherever the microprocessor must control real time events.for examples , motor speed and direction control, and for clock and an events counter.

The timer also appears in the personal computer decoded at ports 40H–43H to do the following:

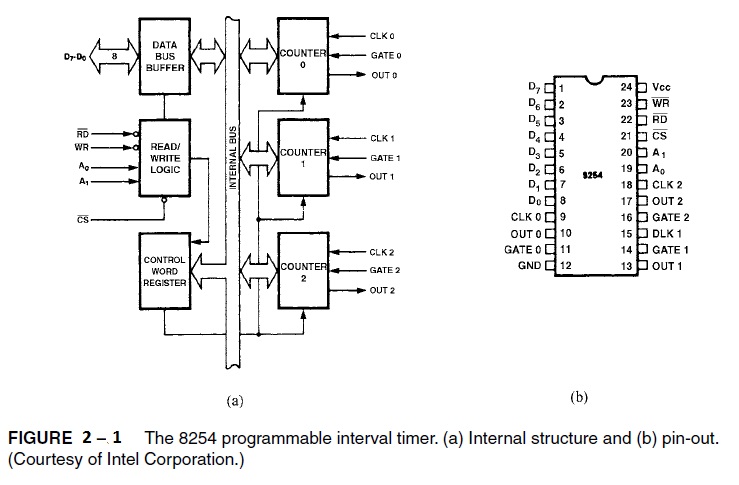
1. Generate a basic timer interrupt that occurs at approximately 18.2 Hz.

2. Cause the DRAM memory system to be refreshed.

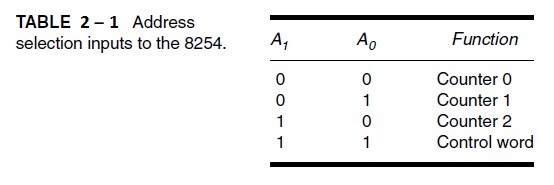
3. Provide a timing source to the internal

**2.4 8253/4 Functional Description**

Figure 2–1 shows the pin-out of the 8254, and a diagram of one counter. Each counter contains a CLK input, a gate input, and OUT pins. The **clock** input is the timing source for each of the internal counters. This input is often connected to the PCLK signal from the microprocessor system bus controller, the **gate** pin controls the timer in some modes, and the OUT pin is where we obtain the output of the timer[[2]](http://en.wikipedia.org/wiki/Diode#cite_note-5).



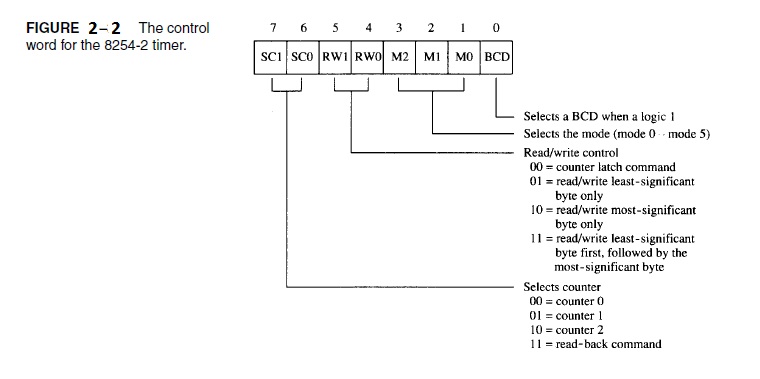
There are many pins that connect to the microprocessor :data bus pins (D7–D0) and address inputs A1, A0. The address inputs select one of four internal registers within the 8254(Table 2–1) [[3]](http://en.wikipedia.org/wiki/Diode#cite_note-5) .



The personal computer contains an 8253 timer or its equivalent, decoded at I/O ports 40H–43H. Timer zero is programmed to generate an 18.2 Hz signal that interrupts the microprocessor at interrupt vector 8 for a clock tick. The tick is often used to time programs and events in DOS. Timer 1 is programmed for 15 μs, which is used on the personal computer to request a DMA action used to refresh the dynamic RAM. Timer 2 is programmed to generate a tone on the personal computer speaker[[2]](http://en.wikipedia.org/wiki/Diode#cite_note-5).

**2.5 Programming the 8254**

Each counter is individually programmed by writing a control word, followed by the initial count (Figure 2–2). The control word allows the programmer to select the counter, mode of operation, and type of operation (read/write). The control word also selects either a binary or BCD count. Each counter may be programmed with a count of 1 to FFFFH.or to 10000 in BCD[[2]](http://en.wikipedia.org/wiki/Diode#cite_note-5).

****

The minimum count of 1 applies to all modes of operation except modes 2 and 3, which have a minimum count of 2.

**3.Procedure**

**3.1 Configuring PIT on PC**

**Setp1: I** Using Dos Command prompt and I start debuging.

**Setp2: I type the following instructions:**

**O 43 B6 //** Program the command register mode3 (show fig 3.1)

**O 42 11 //** send the count 11h to least\_segnificant byte of counter 2

**O 42 11 //** send the count 11h to most\_segnificant byte of counter 2

**O 61 33 //** send 33 to port B (the port that controls the speaker) to enable G2 speaker

**O 61 32 //** send 32 to port B (the port that controls the speaker) to disable G2 speaker

**O 42 55**

**O 42 3F**

**O 43 B0 //** Program the command register mode 0 (show fig 3.2)

**O 42 11 //** send the count 11h to least\_segnificant byte of counter 2

**O 42 11 //** send the count 11h to most\_segnificant byte of counter 2

**O 42 FF**

**O 42 FF**

From the previous code , the output Frequency = = 272Hz. (note 1111H =4369)

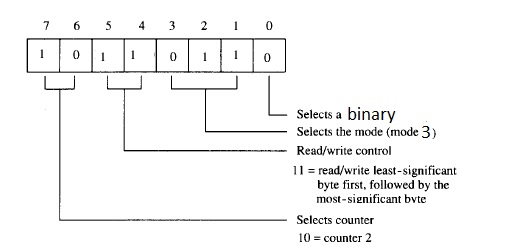
****

Figure 3.1

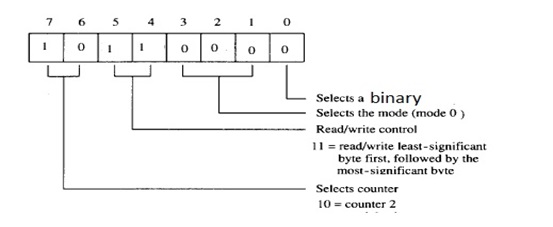


Figure3.2

**3.2 assembly program that enables the speaker with the tone frequency**

**Step1:** I write the following code and save it to an Assembly file(.asm).

The following code enable the speaker with a tone frequency of 5 KH [ the count = 238 ].

To make the output frequency to equal 5KH , 7 KH, 12KH :

= = .

= = .

= = .



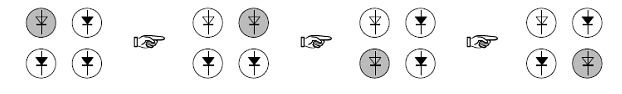
**3.3 Configuring PIT on MDA-8086 Kit**

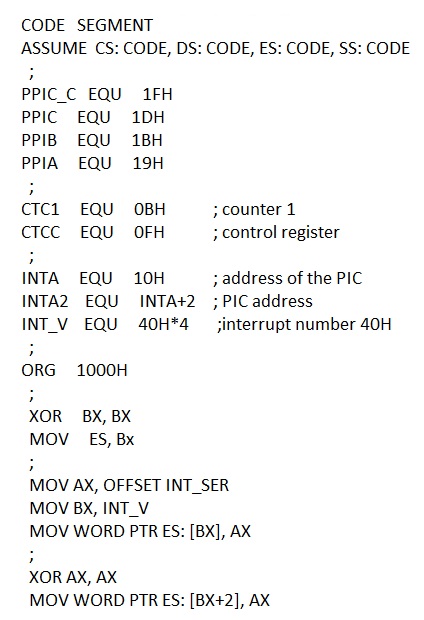
**Srep1:** I connected the P2 using a jumper cap as shown in (Fig3.3).

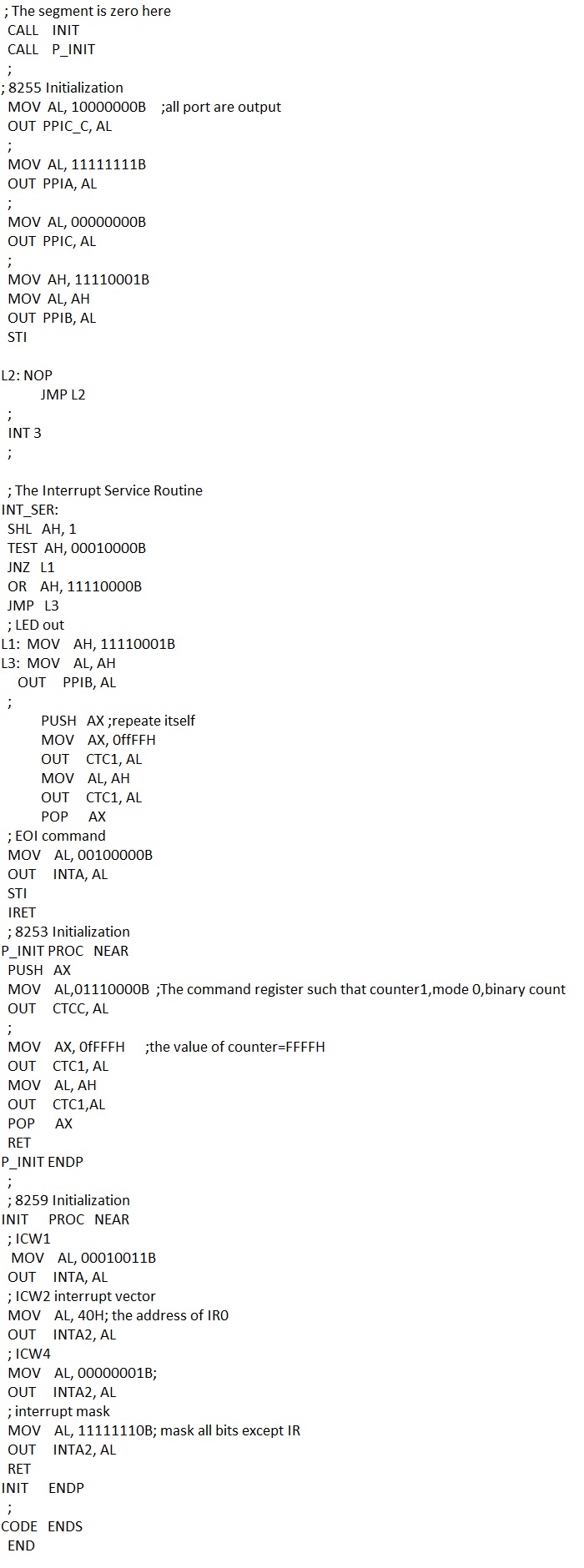


Figure 3.3

**Setp2:** I write the following code and save it to an Assembly file(.asm).I Compile and build this ASM file and execute it on MDA-8086 kit.

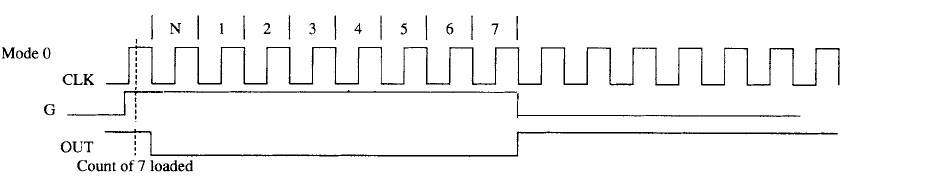
The following code contain ISR to control four LEDs to light, and we used the PIC by connecting the output of counter 1 to IR0 using P2.





🡺The above code use PIT(counter 1) , PIC(IR0) ,P2 and PPI (port B) to light the LEDs.I send the value ffffH to the counter 1 .after the counter reach **FFFFH+1** the interrupt is occur and the next LEDS is light and so on…

When I press the GATE1 the LEDs are stoping continue lighting since the GATE1 connected to the ground.The counter is stoped work when G=0.



**3.4 increase the delay by a factor of 5 times.**

**Srep1:** I connected the P2 using a jumper cap as shown in (Fig3.4).

****

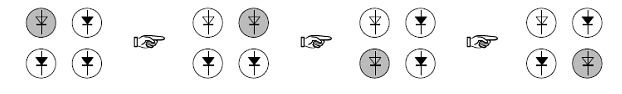
Figure 3.4

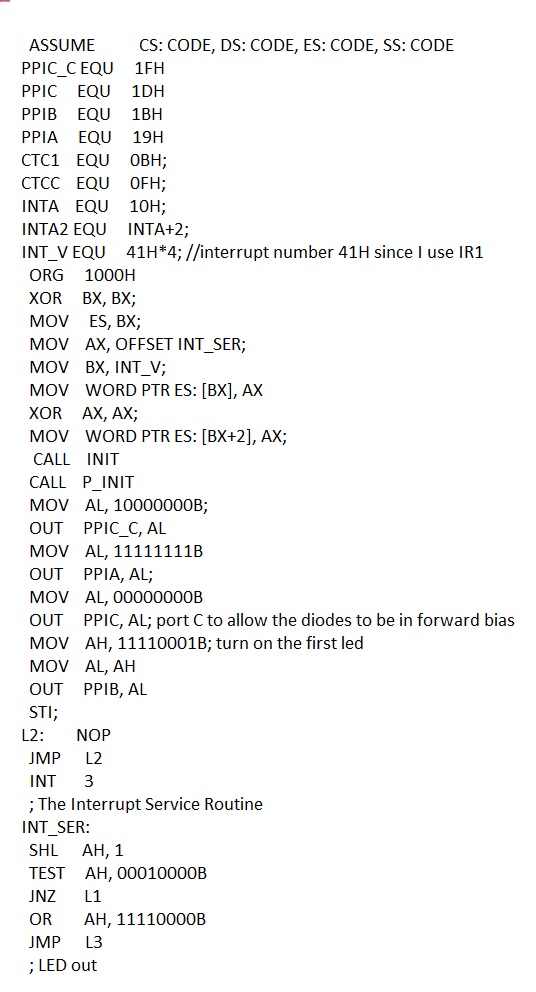
**Setp2:** I write the following code and save it to an Assembly file(.asm).I Compile and build this ASM file and execute it on MDA-8086 kit.

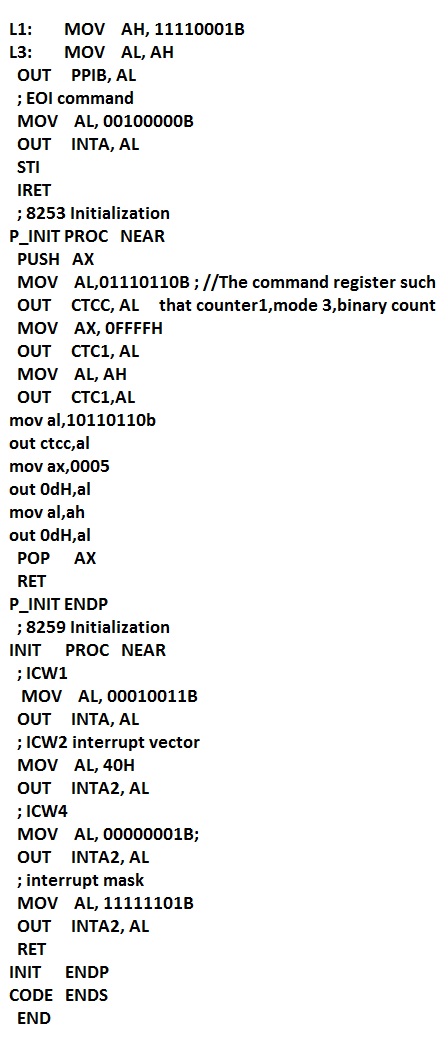
The following code contain ISR to control four LEDs to light(figure below), and i used the PIC by connecting the output of counter 2 to IR1 using P2 and i connecting the output of counter one to the clk of counter2 .

Each time the counter1 count up to 5 the counter 2 incremant by 1

🡺so the total count=FFFFH\*5.



****

****

**4.Conclusion**

In this experiment i learned the Programmable Interval Timer [PIT]and how it work in MDA-8086 Kit . I study the implementation of PIT which consist of three independent counter(timer).I learned how I can use the PIT counter`s to control real-time event .I use the PIT to light the LEDs and I change the speed of lighting by decreasing frequency.

**5.Reference**

[1] http://en.wikipedia.org/wiki/Intel\_8253

[2] The Intel Microprocessors by Barry.B.Bary (Eighth Edition).

[3] Experement#8 lab manual.